Joint Video Team (JVT) of ISO/IEC MPEG & ITU-T VCEG (ISO/IEC JTC1/SC29/WG11 and ITU-T SG16 Q.6)
7th Meeting: Pattaya II, Thailand, 7-14 March, 2003

Document: JVT-G025
Filename: JVT-G025.doc

Title: **Speed up Motion Compensation Interpolation Using On-Chip Memory**

Status: Input Document to JVT

Purpose: Proposal

Author(s) or Contact(s): Lifeng Song
College of Electronic & Information Engineering, South China University of Technology, Guangzhou 510641, China
Tel: 86-20-38457381
Email: slf@21cn.com

Source: South China University of Technology

1. Introduction

The memory access problem of motion compensation interpolation was concerned at the recent JVT meetings. In this proposal a solution for this problem using on-chip memory is presented. 5 sample values for JVT 6-tap interpolation filtering are stored into on-chip memory such as MMX registers of Intel Pentium CPU or L2 memory of TI TMS320C64x DSP. Only one more sample value is required to be loaded from external RAM to accomplish each interpolation. Thus memory bandwidth for motion compensation interpolation is reduced effectively while computational complexity including 32-bit arithmetic length, numbers of multiplication and addition is unchanged. Simulation results on Intel Pentium CPU show that this solution shortens total decoding time up to 22.60% (including B-frames) and 18.23% (excluding B-frames), and the difference between total decoding time of using full 6-tap filter and that of using full 4-tap filter is unobservable. It is proposed to solve the memory access problem using on-chip memory other than modifying the draft text as JVT-D110, JVT-F033 and JVT-F037.

2. Description

2.1. Background

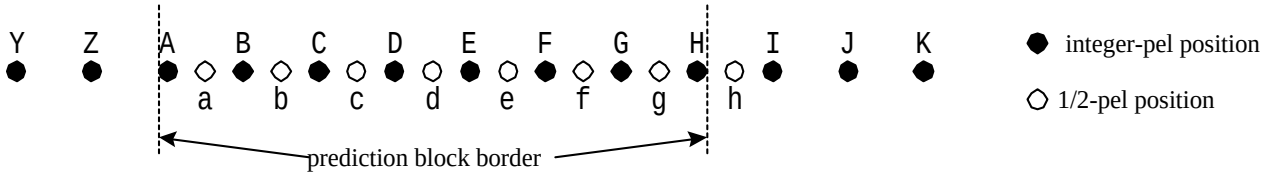
A 6-tap interpolation filter is adopted in the new video coding standard H.264/MPEG-4 AVC for 1/4-pel resolution motion compensated prediction. Thus as filter border overhead 5 extra samples out of a prediction block in each row or column have to be accessed, as showed in Fig 2.1 of JVT-D080. If a macroblock is in 16×16 predictive mode, (16+5) ×(16+5)=441 samples at maximum have to be accessed to interpolate 256 samples at fraction-pel positions. If this macroblock is in full 4×4 predictive mode, 16×(4+5) ×(4+5)=1296 samples at maximum have to be accessed. Each sample of these samples at integer positions is accessed repeatedly 6 times at maximum. In comparison with prediction blocks with the same sizes in P-frames, bi-directional prediction blocks in B-frames require double memory access number to generate forward and backward prediction blocks. So the memory traffic for motion compensation interpolation becomes extremely high. On the other hand, CPU speed is many times faster than memory access speed. For example, Intel Pentium 4 CPU @ 3.06GHz versus DDR-Ⅰ SDRAM @ 400MHz, TI TMS320C64x DSP @ 600MHz versus SDRAM @ 133MHz. In the case of high memory traffic insufficient memory bandwidth results in low running efficiency of high-speed processors. This memory bandwidth problem will show more crucial when high-speed processors with internal parallel architectures such as the HTT (Hyper-Threading Technology) of Intel Pentium 4 CPU and the VelociTI.2 VLIW (Very-Long-Instruction-Word) architecture of TI TMS320C64x DSP are used for real-time playback of HDTV bitstreams.

Many contributions about the memory access problem of motion compensation interpolation were presented at the recent JVT meetings. In JVT-E093 a model for estimation of memory bandwidth as well as the numbers of memory cycles needed for motion compensation is offered. In JVT-D106 a technique called “block boundary mirroring” is presented to limit the number of samples to be accessed for 6-tap

interpolation to that for bilinear interpolation, while still being able to take advantage of the high coding efficiency of 6-tap filter. In JVT-D029 a 4-tap direct filter is presented for bi-directional predictive macroblock type of B-frames, which is equivalent to a subsequent filter containing two sequential 4-tap linear filters. In JVT-D080 the JVT 6-tap filter is used for predictive blocks larger than 8×8 in P-frames while a 4-tap filter is used for smaller blocks in P-frames as well as all blocks in B-frames. In JVT-D110, JVT-F033 and JVT-F037 half-pel motion compensated prediction and bi-linear interpolation are adopted for predictive blocks smaller than 8×8, just as those in MPEG-2; the prediction and interpolation for larger predictive blocks are the same as that described in the JVT draft. Adaptive motion vector coding is also included here. However, these methods reduce both computational load and memory bandwidth at the expense of coding efficiency, especially more loss of rate-distortion performance at smaller QP.

2.2. Solution

In order to obtain the solution based on internal architecture of processors and without modification of draft text, look at 1D 6-tap interpolation at 1/2-pel positions for 8 samples long as below.



Where Y, Z, A, B, C, D, E, F, G, H, I, J, K are the samples at integer-pel positions while a, b, c, d, e, f, g, h, i, j are the samples at 1/2-pel positions. The interpolation is performed as follows

$Y_a=(Y_Y- 5Y_Z+20Y_A+20Y_B- 5Y_C+ Y_D)//32$

$Y_b= (Y_Z- 5Y_A+20Y_B+20Y_C- 5Y_D+ Y_E)//32$

$Y_c= (Y_A- 5Y_B+20Y_C+20Y_D- 5Y_E+ Y_F)//32$

$Y_d= (Y_B- 5Y_C+20Y_D+20Y_E- 5Y_F+ Y_G)//32$

$Y_e= (Y_C- 5Y_D+20Y_E+20Y_F- 5Y_G+ Y_H)//32$

$Y_f= (Y_D- 5Y_E+20Y_F+20Y_G- 5Y_H+ Y_I)//32$

$Y_g= (Y_E- 5Y_F+20Y_G+20Y_H- 5Y_I+ Y_J)//32$

$Y_h= (Y_F- 5Y_G+20Y_H+20Y_I- 5Y_J+ Y_K)//32$

Where Y is the luma sample value. Because on-chip memory space of processors is not enough to store one frame image, integer-pel sample values of reference frames are stored in external RAM as frame buffer. Above equations show the sample values is loaded from external RAM 48 times to interpolate 8 samples at 1D 1/2-pel positions, and the samples D, E, F are accessed 6 times repeatedly respectively.

Table 1 Memory Access Number

Interpolation length (pels)	Access sample number (pels)	Number of loading from external RAM (times)	
		Not using on-chip memory	Using on-chip memory
16	21	96	21
8	13	48	13
4	9	24	9

Regarding the 1st 1/2-pel sample value Y_a , the sample values Y_Y, Y_Z, Y_A, Y_B, Y_C, Y_D are loaded to interpolate Y_a, where Y_Z, Y_A, Y_B, Y_C, Y_D are also used to interpolate the 2nd sample value Y_b, and Y_A, Y_B, Y_C, Y_D are used to interpolate the 3rd sample value Y_c, and so on. If Y_Z, Y_A, Y_B, Y_C, Y_D are stored into on-chip memory at full speed after the 1st interpolation, only part of sample values are loaded from external RAM for the 2nd interpolation and the later interpolations. The other sample values are accessed at full CPU speed from on-chip memory. When the latest 5 32-bit integers for 6-tap interpolation are stored in on-chip memory and also updated after each interpolation, only one more sample value is loaded from external RAM and the other 5 values are loaded from on-chip memory. To interpolate 8 samples sample values are

loaded from external RAM 13 times and from on-chip memory 48-13=35 times, and each sample at integer-pel position is accessed from external RAM once at maximum. The memory access numbers for interpolation length of 16 and 4 are listed in Table 1.

In Intel Pentium CPU with the MMX Technology, 8 64-bit MMX registers can be used to store 5 32-bit integers for 6-tap interpolation. In TI TMS320C64x DSP up to 8M-bit (that is 1024K-byte) Level 2 memory/cache can be flexibly allocated and used as mapped memory to store data. The access speed of these internal registers or memory is just the CPU speed. It is many times faster than that of external RAM. Thus the memory bandwidth is reduced dramatically and the running efficiency of processors is raised.

2.3. Implementation

Let's turn to the implementation of the above solution on Intel Pentium CPU. The subroutine of 1D 6-tap interpolation using MMX registers is written in x86 assembly codes. Some of the assembly codes are listed below.

```
                mov     edi, [input image pointer]
                mov     esi, [output image pointer]
                mov     ecx, [inpterpolation length]
                …
Interpolation:  xor     eax, eax        //clear eax to 0
                mov     al, edi         //load 8-bit image data to al (low 8-bit of eax)
                inc     edi             //point to next image data
                movq    mm5, eax        //move 8-bit image data to mm5
                paddd   mm0, mm5        //low 32-bit addition, mm0<-mm0+mm5
                movd    eax, mm0        //eax<-mm0
                movq    mm0, mm1        //update mm0
                paddd   mm1, mm4        //low 32-bit addition, mm1<-mm1+mm4
                movd    ebx, mm1        //ebx<-mm1
                imul    ebx, 5          //ebx<-5×(mm1+mm4)
                sub     eax, ebx        //eax<-(mm0+mm5)-(mm1+mm4)
                movq    mm1, mm2        //update mm1
                paddd   mm2, mm3        //low 32-bit addition, mm2<-mm2+mm3
                movd    ebx, mm2        //ebx<-mm2
                imul    ebx, 20         //ebx<-20×(mm2+mm3)
                add     eax, ebx        //eax<-(mm0+mm5)-5×(mm1+mm4)+20×(mm2+mm3)
                mov     [esi], eax      //output interpolation result
                add     esi, 4          //point to next output result
                movq    mm2, mm3        //update mm2
                movq    mm3, mm4        //update mm3
                movq    mm4, mm5        //update mm4
                loop    Interpolation   //ecx<-ecx-1, loop until ecx==0
```

Please refer to the decoder software JVT-G025_software.zip for the details. It can be seen from the above codes that some operations are saved as the MMX registers can be used directly for arithmetic/logic operation. 6 sample values of 6-tap interpolation are stored into the low 32 bit of 6 MMX registers mm0~mm5 respectively. In fact the on-chip memory space enough to accommodate 5 32-bit integers (that is 5 MMX registers) is enough for the solution. One more MMX register is use in the above implementation to reduce operation number. The complexity of the above implementation including computational load, 32-bit arithmetic length, numbers of multiplication and addition, etc, is the same as that of the JVT reference software described in the standard draft. Moreover, this implementation does not contain any SIMD operation of MMX, 3Dnow!, SSE and SSE2 on purpose to investigate the effect of the solution. The inline assembly function containing the said assembly codes is incorporated into the JVT software decoder version JM5.0C. This software decoder in JVT-G025_software.zip can be compiled and linked by MS Visual C++ 6.0.

In the current JVT reference software, motion compensation interpolation is based on 4×4 block and its subroutine is called once for each 4×4 block. So this subroutine has to be called multiple times for $N \times M$ prediction blocks larger than 4×4. When on-chip memory is used, the maximal access number at each sample of 4×4 block-based interpolation is two while that of $N \times M$ block-based interpolation is one. The number of loading from external RAM of 4×4 block-based interpolation is also more than that of $N \times M$ block-based interpolation. In order to obtain the best result, the JVT decoder software is modified so that the interpolation is performed adaptively to the size of prediction blocks which is indicated by 16×16 macroblock type and 8×8 sub-macroblock type. That is to say the subroutine of motion compensation interpolation is called once definitely for each $N \times M$ prediction block. The 2D interpolation length $N \times M$ can be decided by the current macroblcok type or sub-macroblcok type for all the prediction blocks except those in the temporal/spatial direct mode in B-frames. Concerning the blocks in the 16×16 temporal/spatial direct mode or 8×8 temporal/spatial direct mode in B-frames, the size of the current block is decided by the macroblcok type or sub-macroblcok type of the co-located macroblcok or sub-macroblock in the nearest backward reference frame. Thus the macroblock types and sub-macroblock types of each I-frames or P-frame have to be stored for the interpolation of the prediction blocks in the direct mode in B-frames.

3. Simulation Results

Before comparison of decoding speed, several bitstreams have been generated under the following encoding condition. The decoder based on the JM5.0C decoder using the MMX registers to accelerate motion compensation interpolation decoded these JM5.0C bitstreams perfectly without any mismatch. Simulation ran on a laptop computer with a Intel Celeron CPU (Tualatin) @1.2GHz and 100MHz SDRAM. The Simulation results are in Table 3 and Table 4, where JM5.0C means the JVT reference software decoder JM5.0C, MMX means the decoder using the MMX registers as the solution for the memory bandwidth problem.

Table 2 Encoding Condition

Codec	JM5.0C
Sequence	CIF: Mobile & Calendar, Paris, Tempete, Foreman PAL SD: Canoa, F1 Car NTSC SD: Football, Waterfall
GOP	(1)IPPPPP… (2)IBBPBB… (I-Picture: First Frame Only)
QP	$QP_I = QP_P$ = 20,24,28,32,36,40 $QP_B = QP_I + 2$
Entropy Coding	CABAC
RD Optimization	On
Hadamard	Yes
Motion Compensation Accuracy	1/4-pel
Ref. Frames	5
MV Search Range	±16
Restricted Search Range	2

As see the simulation results, when B frame are not included, the total decoding time of the decoder using the MMX registers is shortened by 2.95%~18.23% in comparison with that of the decoder without using the MMX registers. It is shortened by 4.60%~22.60% when B frames are included.

Table 3 Simulation results for GOP=IPPPPP

Sequence	Number of frames	Decoder	Total decoding time (second)					
			QP=20	QP=24	QP=28	QP=32	QP=36	QP=40
Mobile & Calendar	300 frames CIF (352×288)	JM5.0C	64.35	61.29	59.00	57.03	55.66	54.70
		MMX	56.03	53.07	50.64	47.60	45.66	44.72
		Shorten	12.93%	13.42%	14.18%	16.54%	17.97%	18.23%
Paris	300 frames CIF (352×288)	JM5.0C	48.75	47.19	46.10	45.75	44.48	43.55
		MMX	45.72	43.97	43.12	42.28	41.45	40.91
		Shorten	6.22%	6.81%	6.45%	7.60%	6.83%	6.07%
Tempete	260 frames CIF (352×288)	JM5.0C	55.69	53.41	50.99	49.45	47.82	46.37
		MMX	47.46	45.32	43.01	41.09	39.42	38.35
		Shorten	14.78%	15.14%	15.65	16.91%	17.56%	17.29%
Foreman	300 frames CIF (352×288)	JM5.0C	59.64	56.83	55.37	54.05	52.38	49.53
		MMX	50.36	47.75	45.93	44.40	43.32	42.60
		Shorten	15.56%	15.99%	17.06%	17.85%	17.30%	13.98%
Canoa VQEG src05	220 frames PAL SDTV (720×576)	JM5.0C	298.66	294.85	288.99	286.95	284.64	284.38
		MMX	285.32	281.90	273.61	270.07	267.74	265.51
		Shorten	4.47%	4.39%	5.32%	5.88%	5.94%	6.64%
F1 Car VQEG src06	220 frames PAL SD (720×576)	JM5.0C	296.33	291.27	289.08	285.50	280.86	278.01
		MMX	287.59	280.50	277.47	272.45	267.69	263.83
		Shorten	2.95%	3.70%	4.02%	4.57%	4.69%	5.10%
Football VQEG src19	260 frames NTSC SD (720×480)	JM5.0C	271.02	268.37	261.93	256.99	250.29	241.08
		MMX	256.47	250.09	243.50	237.27	232.93	228.44
		Shorten	5.37%	6.81%	7.04%	7.67%	6.94%	5.24%
Waterfall VQEG src18	260 frames NTSC SD (720×480)	JM5.0C	290.19	281.45	273.72	263.54	253.81	238.07
		MMX	260.54	245.38	238.52	231.68	228.45	224.68
		Shorten	10.31%	12.82%	12.86%	12.09%	9.99%	5.62%

Table 4 Simulation results for GOP=IBBPBB

Sequence	Number of frames	Decoder	Total decoding time (second)					
			QP_I=20	QP_I =24	QP_I =28	QP_I =32	QP_I =36	QP_I =40
Mobile & Calendar	298 frames CIF (352×288)	JM5.0C	70.39	68.51	66.81	65.13	61.55	59.87
		MMX	58.26	55.32	53.46	50.44	47.64	46.48
		Shorten	17.24%	19.25%	19.97%	22.56%	22.60%	22.36
Paris	298 frames CIF (352×288)	JM5.0C	55.07	54.25	53.26	52.19	51.30	51.04
		MMX	50.93	49.53	48.80	48.26	47.65	47.74
		Shorten	7.51%	8.69%	8.37%	7.54%	7.12%	6.46%
Tempete	259 frames CIF (352×288)	JM5.0C	60.79	58.36	55.83	53.35	50.88	47.91
		MMX	49.24	46.56	44.34	43.09	42.20	41.81
		Shorten	18.99%	20.22%	20.57%	19.22%	17.06%	12.72%
Foreman	298 frames CIF (352×288)	JM5.0C	64.89	62.66	60.91	59.45	56.58	53.83
		MMX	52.24	49.75	48.29	47.02	47.18	46.67
		Shorten	19.50%	20.60%	20.73%	20.90%	16.61%	13.30%
Canoa VQEG src05	220 frames PAL SDTV (720×576)	JM5.0C	294.05	290.14	287.36	287.48	289.47	289.48
		MMX	279.89	274.45	270.11	268.64	268.68	269.39
		Shorten	4.81%	5.41%	6.00%	6.55%	7.18%	6.94%
F1 Car VQEG src06	220 frames PAL SD (720×576)	JM5.0C	300.62	298.59	298.26	298.04	295.82	292.28
		MMX	286.79	282.34	279.60	277.72	275.55	274.80
		Shorten	4.60%	5.44%	6.26%	6.82%	6.85%	5.98%
Football VQEG src19	259 frames NTSC SD (720×480)	JM5.0C	270.85	264.80	261.92	259.72	254.89	253.49
		MMX	250.24	244.64	238.65	238.89	238.68	239.19
		Shorten	7.61%	7.61%	8.88%	8.02%	6.36%	5.64%
Waterfall VQEG src18	259 frames NTSC SD (720×480)	JM5.0C	316.75	299.97	288.24	270.84	261.56	257.38
		MMX	266.69	253.17	246.07	242.70	243.36	243.97
		Shorten	15.81%	15.60%	14.63%	10.39%	6.96%	5.21%

4. Summary

A proposal a solution for this problem using on-chip memory is presented. 5 sample values for JVT 6-tap interpolation filtering are stored into on-chip memory such as MMX registers of Intel Pentium CPU or L2 memory of TI TMS320C64x DSP. Only one more sample value is required to be loaded from external RAM to accomplish each interpolation. Thus memory bandwidth for motion compensation interpolation is reduced effectively while computational complexity including 32-bit arithmetic length, numbers of multiplication and addition is unchanged. Simulation results on Intel Pentium CPU show that this solution shortens total decoding time up to 22.60% (including B-frames) and 18.23% (excluding B-frames), and the difference between total decoding time of using full 6-tap filter and that of using full 4-tap filter is unobservable. It is proposed to solve the memory access problem using on-chip memory other than modifying the draft text as JVT-D110, JVT-F033 and JVT-F037.

(Append for Proposal Documents)

JVT Patent Disclosure Form

International Telecommunication Union
Telecommunication Standardization Sector

International Organization for Standardization

International Electrotechnical Commission

Joint Video Coding Experts Group - *Patent Disclosure Form*

(Typically one per contribution and one per Standard | Recommendation)

Please send to:

JVT Rapporteur Gary Sullivan, Microsoft Corp., One Microsoft Way, Bldg. 9, Redmond WA 98052-6399, USA
Email (preferred): Gary.Sullivan@itu.int Fax: +1 425 706 7329 (+1 425 70MSFAX)

This form provides the ITU-T | ISO/IEC Joint Video Coding Experts Group (JVT) with information about the patent status of techniques used in or proposed for incorporation in a Recommendation | Standard. JVT requires that all technical contributions be accompanied with this form. *Anyone* with knowledge of any patent affecting the use of JVT work, of their own or of any other entity ("third parties"), is strongly encouraged to submit this form as well.

This information will be maintained in a "living list" by JVT during the progress of their work, on a best effort basis. If a given technical proposal is not incorporated in a Recommendation | Standard, the relevant patent information will be removed from the "living list". The intent is that the JVT experts should know in advance of any patent issues with particular proposals or techniques, so that these may be addressed well before final approval.

This is not a binding legal document; it is provided to JVT for information only, on a best effort, good faith basis. Please submit corrected or updated forms if your knowledge or situation changes.

This form is *not* a substitute for the *ITU ISO IEC Patent Statement and Licensing Declaration*, which should be submitted by Patent Holders to the ITU TSB Director and ISO Secretary General before final approval.

Submitting Organization or Person:	
Organization name	**South China University of Technology**
Mailing address	College of Electronic & Information Engineering, South China University of Technology, Guangzhou 510641
Country	**China**
Contact person	Lifeng Song
Telephone	86-20-38457381
Fax	
Email	slf@21cn.com
Place and date of submission	Pattaya, Thailand on Mar 03, 2003
Relevant Recommendation | Standard and, if applicable, Contribution:	
Name (ex: "JVT")	**JVT**
Title	**Speed up Motion Compensation Interpolation Using On-Chip Memory**
Contribution number	**JVT-G025**

(Form continues on next page)

Disclosure information – Submitting Organization/Person (choose one box)

☒ 2.0 The submitter is not aware of having any granted, pending, or planned patents associated with the technical content of the Recommendation | Standard or Contribution.

or,

The submitter (Patent Holder) has granted, pending, or planned patents associated with the technical content of the Recommendation | Standard or Contribution. In which case,

☐ 2.1 The Patent Holder is prepared to grant – on the basis of reciprocity for the above Recommendation | Standard – a free license to an unrestricted number of applicants on a worldwide, non-discriminatory basis to manufacture, use and/or sell implementations of the above Recommendation | Standard.

☐ 2.2 The Patent Holder is prepared to grant – on the basis of reciprocity for the above Recommendation | Standard – a license to an unrestricted number of applicants on a worldwide, non-discriminatory basis and on reasonable terms and conditions to manufacture, use and/ or sell implementations of the above Recommendation | Standard.

Such negotiations are left to the parties concerned and are performed outside the ITU | ISO/IEC.

☐ 2.2.1 The same as box 2.2 above, but in addition the Patent Holder is prepared to grant a "royalty-free" license to anyone on condition that all other patent holders do the same.

☐ 2.3 The Patent Holder is unwilling to grant licenses according to the provisions of either 2.1, 2.2, or 2.2.1 above. In this case, the following information must be provided as part of this declaration:

- patent registration/application number;
- an indication of which portions of the Recommendation | Standard are affected.
- a description of the patent claims covering the Recommendation | Standard;

*In the case of any box **other than 2.0** above, please provide the following:*

Patent number(s)/status _____

Inventor(s)/Assignee(s) _____

Relevance to JVT _____

Any other remarks: _____

(please provide attachments if more space is needed)

(form continues on next page)

Third party patent information – fill in based on your best knowledge of relevant patents granted, pending, or planned by other people or by organizations other than your own.

Disclosure information – Third Party Patents (choose one box)

☒ 3.1 The submitter is not aware of any granted, pending, or planned patents *held by third parties* associated with the technical content of the Recommendation | Standard or Contribution.

☐ 3.2 The submitter believes third parties may have granted, pending, or planned patents associated with the technical content of the Recommendation | Standard or Contribution.

For box 3.2, please provide as much information as is known (provide attachments if more space needed) - JVT will attempt to contact third parties to obtain more information:

3rd party name(s) _____

Mailing address _____
Country _____
Contact person _____
Telephone _____
Fax _____
Email _____
Patent number/status _____
Inventor/Assignee _____
Relevance to JVT _____

Any other comments or remarks: